OBJECT ORIENTED PROGRAMMING WITH JAVA 8– LAB 7

1. Write a program to calculate the square value of any number given by the user. Add an exception handling block to check whether the user enter letters instead of numbers.

Ans=

Code-

```
import java.util.InputMismatchException;
import java.util.Scanner;
public class Square
{
        public static void main(String[] args)
        {
         calculateSquare();
        }

        public static void calculateSquare()
        {
         Scanner P = new Scanner(System.in);

         try
        {
         System.out.print("Enter a Number: ");
         int number = P.nextInt();
         int square = (int)Math.pow(number, 2);
         System.out.println("The square of: " + number + " is: " + square);
        }
         catch(InputMismatchException e)
        {
         System.out.println("Not an integer input and Error is " + e);
        }
        }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA\Day 7 & Lab7>javac Square.java

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA\Day 7 & Lab7>java Square
Enter a Number: a
Not an integer input and Error is java.util.InputMismatchException

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA\Day 7 & Lab7>java Square
Enter a Number: 7
The square of: 7 is: 49

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA\Day 7 & Lab7>
```

2. Create an integer array of size n and read the elements from the user. Add an exception handling block to print the value at nth position of the array.

Ans=

Code-

```
import java.util.Scanner;

public class ArrayExceptionHandling
{
        public static void main(String[] args)
        {
                Scanner P = new Scanner(System.in);

                System.out.print("Enter the size of the array: ");
                int n = P.nextInt();

                int[] arr = new int[n];

                for (int i = 0; i < n; i++)
                {
                System.out.print("Enter element at position " + i + ": ");
                arr[i] = P.nextInt();
                }

                 System.out.print("Enter the position to print (0 to " + (n - 1) + "): ");
                 int position = P.nextInt();

                 try {
                       System.out.println("Value at position " + position + ": " + arr[position]);
                     }
                       catch (ArrayIndexOutOfBoundsException e)
                     {
                       System.err.println("Invalid position. Please enter a valid position within the array size.");
                     }
        }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>javac ArrayExceptionHandling.
java

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java ArrayExceptionHandling
Enter the size of the array: 7
Enter element at position 0: 1
Enter element at position 1: 2
Enter element at position 2: 3
Enter element at position 3: 4
Enter element at position 4: 5
Enter element at position 5: 6
Enter element at position 6: 7
Enter the position to print (0 to 6): 8
Invalid position. Please enter a valid position within the array size.

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>
```

3. Write a program to read a string and convert to integer using try catch block.

Ans=

Code-

```
import java.util.Scanner;

public class StringToIntegerConversion
{
        public static void main(String[] args)
        {
         Scanner P = new Scanner(System.in);

         System.out.print("Enter a string to convert to an integer: ");
         String userInput = P.nextLine();

                 try
                {
                 int convertedInteger = Integer.parseInt(userInput);
                 System.out.println("Successfully converted to an integer: " + convertedInteger);
                }
                 catch (NumberFormatException e)
                {
                 System.err.println("Invalid input. The provided string is not a valid integer.");
                }
        }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>javac StringToIntegerConversion.java

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java StringToIntegerConversion
Enter a string to convert to an integer: 7
Successfully converted to an integer: 7

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>
```

4. Create a class named MarkProcess to process the marks with following members:

a. Data Members

i. regno

ii. marks

b. Function members

i. Constructor to accept all values

ii. validation()- checking marks < 0 and throwing a user defined

exception named IllegalMarkException.

iii. result()- declaring PASS if marks>=40 and FAIL otherwise

Create a user defined exception class named IllegalMarkException and handle with the message 'Illegal Mark'. Write a main() method that will create an object of type MarkProcess and call the methods in it to declare the result only for valid marks.

Ans=

Code- Part 1,

```
class IllegalMarkException extends Exception
{
        public IllegalMarkException(String message)
        {
         super(message);
        }
}

class MarkProcess
        {
         int regno;
         int marks;

        public MarkProcess(int regno, int marks)
        {
         this.regno = regno;
         this.marks = marks;
        }

        public void validation() throws IllegalMarkException
        {
         if (marks < 0)
                {
                 throw new IllegalMarkException("Illegal Mark");
                }
        }
```

Part 2,

```
        public void result()
        {
        if (marks >= 40)
                {
                 System.out.println("Student with registration number " + regno + " has passed.");
                }
        else
                {
                 System.out.println("Student with registration number " + regno + " has failed.");
                }
        }

        public static void main(String[] args)
        {
         try
                {
                MarkProcess student1 = new MarkProcess(101, 85);
                student1.validation();
                student1.result();

                MarkProcess student2 = new MarkProcess(102, -10);
                student2.validation();
                student2.result();
                }
        catch (IllegalMarkException e)
                {
                 System.out.println("Exception: " + e.getMessage());
                }
        }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>javac MarkProcess.java

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java MarkProcess
Student with registration number 101 has passed.
Exception: Illegal Mark

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>
```

5. Write a program to read a binary number and convert it to decimal number. Throw user defined exception named InvalidBinaryException if the number entered is not binary and handle with the message ‘Not a valid Binary number’.

Ans=

Code- Part 1,

```
class InvalidBinaryException extends Exception
{
         public InvalidBinaryException(String message)
        {
         super(message);
        }
}

public class BinaryToDecimalConverter
{
        public static int convertBinaryToDecimal(String binary) throws InvalidBinaryException
        {
        int decimal = 0;
        int power = 0;

        for (int i = binary.length() - 1; i >= 0; i--)
        {
            char bit = binary.charAt(i);
            if (bit != '0' && bit != '1')
                {
                 throw new InvalidBinaryException("Not a valid Binary number");
                }

            int bitValue = bit - '0';
            decimal += bitValue * Math.pow(2, power);
            power++;
        }

        return decimal;
        }
```

Part 2,

```
        public static void main(String[] args)
        {
         try
                {
                 String binaryInput = "101010";

                 int decimalResult = convertBinaryToDecimal(binaryInput);

                 System.out.println("Decimal equivalent of " + binaryInput + " is: " + decimalResult);
                }
        catch (InvalidBinaryException e)
                {
                 System.err.println("Exception: " + e.getMessage());
                }
        }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>javac BinaryToDecimalConverter.java

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java BinaryToDecimalConverter
Decimal equivalent of 101010 is: 42

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>
```

6. Use MySQL to create database named company and table named emp.

emp (id integer primary key, name varchar(25), age int , salary int);

Insert some rows in emp table

insert into emp values(……);

Write separate methods in a Java application to do the following tasks:

a) Select entire content of emp table and display on screen

b) Display the name and salary of a particular employee whose id is given

c) Insert new row in the emp table

d) Update salary of a particular employee whose id is given

e) Delete the details of an employee whose id is given

f) Select the details of employees whose age is greater than a particular value

Use Statement for a) and b) and use PreparedStatement for c), d), e) and f).

Ans=

```
mysql> use company;
Database changed
mysql> CREATE TABLE IF NOT EXISTS emp (id INT PRIMARY KEY,name VARCHAR(25),age INT, salary INT);
Query OK, 0 rows affected (0.02 sec)

mysql> INSERT INTO emp (id, name, age, salary) VALUES (1, 'Rushi Tapdiya', 30, 60000);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO emp (id, name, age, salary) VALUES (2, 'Ramesh Pandit', 35, 70000);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO emp (id, name, age, salary) VALUES (3, 'Mahesh Pradhan', 28, 55000);
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM emp
    -> ;
+----+---------------+------+--------+
| id | name          | age  | salary |
+----+---------------+------+--------+
|  1 | Rushi Tapdiya |   30 |  60000 |
|  2 | Ramesh Pandit |   35 |  70000 |
|  3 | Mahesh Pradhan |   28 |  55000 |
+----+---------------+------+--------+
3 rows in set (0.00 sec)
```

Code- Part 1,

```
import java.sql.*;

public class EmployeeDatabaseApp
        {
         private static final String JDBC_URL = "jdbc:mysql://localhost:3306/company";
         private static final String USER = "root";
         private static final String PASSWORD = "Pratik@17#mysql";

        public static void main(String[] args)
        {
         try {
                Class.forName("com.mysql.cj.jdbc.Driver");
                Connection connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);

                selectAllEmployees(connection);

                displayEmployeeDetails(connection, 1);

                insertEmployee(connection, 4, "Vedant Pandit", 30, 60000);

                updateEmployeeSalary(connection, 2, 70000);

                deleteEmployee(connection, 3);

                selectEmployeesByAge(connection, 25);

             }
         catch (Exception e) {
                             e.printStackTrace();
                            }
        }
```

Part 2,

```
         private static void selectAllEmployees(Connection connection) throws SQLException
         {
          Statement statement = connection.createStatement();
          ResultSet resultSet = statement.executeQuery("SELECT * FROM emp");

          while (resultSet.next())
         {
             System.out.println("ID: " + resultSet.getInt("id") +
                     ", Name: " + resultSet.getString("name") +
                     ", Age: " + resultSet.getInt("age") +
                     ", Salary: " + resultSet.getInt("salary"));
         }

         resultSet.close();
         statement.close();
    }
```

Part 3,

```
private static void displayEmployeeDetails(Connection connection, int employeeId) throws SQLException
    {
     Statement statement = connection.createStatement();
     ResultSet resultSet = statement.executeQuery("SELECT name, salary FROM emp WHERE id = " + employeeId);

     if (resultSet.next())
            {
             System.out.println("Employee Name: " + resultSet.getString("name") + ", Salary: " + resultSet.getInt("salary"));
            }
    else
            {
             System.out.println("Employee with ID " + employeeId + " not found.");
            }

     resultSet.close();
     statement.close();
    }
```

Part 4,

```
   private static void insertEmployee(Connection connection, int id, String name, int age, int salary) throws SQLException
       {
       String query = "INSERT INTO emp (id, name, age, salary) VALUES (?, ?, ?, ?)";
       PreparedStatement preparedStatement = connection.prepareStatement(query);

       preparedStatement.setInt(1, id);
       preparedStatement.setString(2, name);
       preparedStatement.setInt(3, age);
       preparedStatement.setInt(4, salary);

       preparedStatement.executeUpdate();
       preparedStatement.close();
       }
```

Part 5,

```
private static void updateEmployeeSalary(Connection connection, int employeeId, int newSalary) throws SQLException
    {
     String query = "UPDATE emp SET salary = ? WHERE id = ?";
     PreparedStatement preparedStatement = connection.prepareStatement(query);

     preparedStatement.setInt(1, newSalary);
     preparedStatement.setInt(2, employeeId);

     int rowsAffected = preparedStatement.executeUpdate();
     if (rowsAffected > 0)
            {
             System.out.println("Salary updated successfully for employee with ID " + employeeId);
            }
    else
            {
             System.out.println("No employee found with ID " + employeeId);
            }

    preparedStatement.close();
    }
```

Part 6,

```
private static void deleteEmployee(Connection connection, int employeeId) throws SQLException
    {
     String query = "DELETE FROM emp WHERE id = ?";
     PreparedStatement preparedStatement = connection.prepareStatement(query);

     preparedStatement.setInt(1, employeeId);

     int rowsAffected = preparedStatement.executeUpdate();
     if (rowsAffected > 0)
           {
            System.out.println("Employee with ID " + employeeId + " deleted successfully");
           }
     else
           {
            System.out.println("No employee found with ID " + employeeId);
           }

     preparedStatement.close();
    }
```

Part 7,

```
    private static void selectEmployeesByAge(Connection connection, int ageThreshold) throws SQLException
       {
        String query = "SELECT * FROM emp WHERE age > ?";
        PreparedStatement preparedStatement = connection.prepareStatement(query);

        preparedStatement.setInt(1, ageThreshold);
        ResultSet resultSet = preparedStatement.executeQuery();

        while (resultSet.next())
              {
               System.out.println("ID: " + resultSet.getInt("id") +
                  ", Name: " + resultSet.getString("name") +
                  ", Age: " + resultSet.getInt("age") +
                  ", Salary: " + resultSet.getInt("salary"));
              }

        resultSet.close();
        preparedStatement.close();
       }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java EmployeeDatabaseApp
Tue Oct 17 21:58:46 IST 2023 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26
+ and 5.7.6+ requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SS
L the verifyServerCertificate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide
truststore for server certificate verification.
ID: 1, Name: Rushi Tapdiya, Age: 30, Salary: 60000
ID: 2, Name: Ramesh Pandit, Age: 35, Salary: 70000
ID: 3, Name: Mahesh Pradhan, Age: 28, Salary: 55000
Employee Name: Rushi Tapdiya, Salary: 60000
Salary updated successfully for employee with ID 2
Employee with ID 3 deleted successfully
ID: 1, Name: Rushi Tapdiya, Age: 30, Salary: 60000
ID: 2, Name: Ramesh Pandit, Age: 35, Salary: 70000
ID: 4, Name: Vedant Pandit, Age: 30, Salary: 60000

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>
```

7. Create a stored procedure empproc in the database from MySQL.

Use the following command:

create procedure empproc(in eid int , out ename varchar(25))

begin

select name into ename from emp where id =eid;

end

Write a Java application which calls the above procedure.

Ans=

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE empproc(INOUT eid INT, OUT ename VARCHAR(25))
    -> BEGIN
    -> SELECT name INTO ename FROM emp WHERE id = eid;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
```

Code- Part 1,

```
import java.sql.CallableStatement;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Types;

public class CallStoredProcedureExample
        {
         private static final String JDBC_URL = "jdbc:mysql://localhost:3306/company";
         private static final String USER = "root";
         private static final String PASSWORD = "Pratik@17#mysql";

         public static void main(String[] args)
        {
                try
                   {
                    Class.forName("com.mysql.cj.jdbc.Driver");
                    Connection connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);

                    int employeeId = 1;
                    String employeeName = callStoredProcedure(connection, employeeId);

                    System.out.println("Employee name for ID " + employeeId + ": " + employeeName);

                    connection.close();
                   }
                catch (Exception e)
                        {
                         e.printStackTrace();
                        }
        }
```

Part2,

```
        private static String callStoredProcedure(Connection connection, int employeeId)
        {
            try
                {
                 CallableStatement callableStatement = connection.prepareCall("{call empproc(?, ?)}");
                 callableStatement.setInt(1, employeeId);
                 callableStatement.registerOutParameter(2, Types.VARCHAR);

                 callableStatement.execute();

                 String employeeName = callableStatement.getString(2);

                 callableStatement.close();

                 return employeeName;
                }
           catch (Exception e)
                {
                 e.printStackTrace();
                 return null;
                }
        }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java CallStoredProcedureExample
Tue Oct 17 22:23:41 IST 2023 WARN: Establishing SSL connection without server's identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26
+ and 5.7.6+ requirements SSL connection must be established by default if explicit option isn't set. For compliance with existing applications not using SS
L the verifyServerCertificate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide
truststore for server certificate verification.
Employee name for ID 1: Rushi Tapdiya
```